**COMP1411 (Spring 2021) Introduction to Computer Systems**

Individual Assignment 2 Due Date: 10:00am, 5th April, 2021

|  |  |
| --- | --- |
| *Name* |  |
| *Student number* |  |

**Question 1**. [30 marks]

In this question, we use the Y86-64 instruction set (please refer to Lecture 4-6).

**1(a) Write** the machine code encoding of the assembly instruction “rmmovq %rcx,

0x430(%rbx)”. [10 marks]

Please write the bytes of the machine code in hex-decimal form, i.e., using two letters to represent the content of one byte. You are allowed to leave spaces between adjacent bytes for better readability.

*Answer*:

40 13 30 04 00 00 00 00 00 00

**1(b)** Consider the execution of the instruction “rmmovq %rcx, 0x430(%rbx)”. Assume that for now, the data in register %rcx is 10, the data in register %rbx is 0x100, just before executing this instruction, the value of PC is 0x120.

**Describe** the steps done in the following stages: Fetch, Decode, Execute, Memory, Write Back, PC update, by filling in the blanks in the table below.

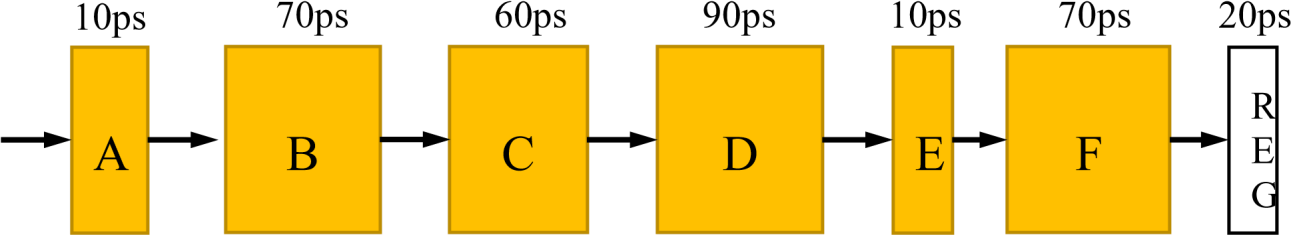
Note that you are required to fill in the generic form of each step in the second column; and in the third column, fill in the steps for the instruction “rmmovq %rcx, 0x430(%rbx)” with the above given values. If you think there should not be a step in some stage, just leave the blanks unfilled.

The symbol “←” means reading something from the right side and assign the value to the left side. X:Y means assign the highest 4 bits of a byte to X, and assign the lowest 4 bits of the byte to Y. [20 marks] *Answer*:

|  |  |  |
| --- | --- | --- |
| **Stages** | **rmmovq rA, D(rB)** | **rmmovq %rcx, 0X430(%rbx)** |
| Fetch | icode: ifun ← \_M1[PC]\_\_ rA:rB ← \_M1[PC+1]\_\_\_ valC ← \_\_M8[PC+2]\_\_ valP ← \_PC+10\_ | icode: ifun ← \_M1[0x120]  =4:0  rA:rB ←  \_M1[0x121]=1:3  valC ←  \_\_M8[0x122]=1072 valP ←  \_0x120+10=0x12a |
| Decode | valA ← \_\_R[rA]\_\_\_ valB  ← \_\_R[rB]\_\_\_ | valA ← \_R[%rcx]=10\_\_ valB ←  \_R[%rbp]=0x100\_\_ |
| Execute | valE ← \_\_va1B+va1C\_ | valE ← \_0x100+0x430 = 1328\_ |
| Memory | M8[\_va1E\_] ← \_\_va1A\_\_\_ | M8[1328] ←10\_\_ |
| Write back | \_\_\_\_\_\_\_\_\_ ← \_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_ ← \_\_\_\_\_\_\_\_\_ |
| PC update | PC ← \_\_va1P\_\_ | PC ← 0x12A\_\_\_ |

**Question 2**. [30 marks]

Suppose a combinational logic is implemented by 6 serially connected components named from A to F. The whole computation logic can be viewed as an instruction. The number on each component is the time delay spent on this component, in time unit ps, where 1ps = 10-12 second. Operating each register will take 20ps.



Thoughput is defined as how many instructions can be executed on average in one second for a pipeline, and the unit of throughput is IPS, instructions per second. Latency refers to the time duration starting from the very first component and ending with the last register operation finished, the time unit for latency is ps.

For throughput, please write the result in the form X.XX \* 10Y IPS, where X.XX means one digit before the dot and two fractional digits after the dot, and Y is the exponent.

**2(a)** Insert two registers (not including the register in the figure) making the computation logic a

3-stage pipeline design that has the maximal throughput. [15 marks] •

Please answer where the two registers are inserted respectively.

• Please compute the throughput and latency for your pipeline design, with steps.

Answer:

I think total 5 component,3 process.

T = S1+S2+S3\*3

So the Throughput = 1/(160\*10^-12)= 6.25\*(10^9) (IPS)

Latency = (140+20)\*3 = 480 (ps)

**2(b)** Insert three registers (not including the register in the figure) making the computation logic a 4-stage pipeline design that has the maximal throughput. [15 marks] • Please answer where the two registers are inserted respectively.

• Please compute the throughput and latency for your piepeline design, with steps.

Answer:

Graphs AB|C|D|E|F

Th = 1/((90+20)\*10^-12)= 9.09\*(10^9) (IPS)

Latency = (90+20)\*4 = 440 (ps)

**Question 3**. [40 marks]

The following byte sequence is the machine code of a program function compiled with the

Y8664 instruction set (refer to Lecture 6). The memory address of the first byte is 0x100. Note that the byte sequence is written in hex-decimal form, i.e., each number/letter is one hex-decimal number representing 4 binary bits, and two numbers/letters represent one byte.

**30|F3|14|00|00|00|00|00|00|00|63|00|30|F2|01|00|00|00|00|00|00|00|7**

**0|23|01|00|00|00|00**

**00|00|60|30|61|23|62|33|76|1F|10|00|00|00|00|00|00|90**

Please write out the assembly instructions (in Y86-64 instruction set) corresponding to the machine codes given by the above bytes sequence, and explain what this program function is computing.

Answer:

**irmovq 0x14,%rbx**

**xorq %rax,%rax (src,Dest,D=D^s)**

**irmovq 0x01,%rdx**

**jmp**

**addq %rbx1,%rax** （**D=D+S**） **subq %rdx,%rbx** (**D=D-S) andq %rbx,%rbx** （**D=D&S**） **jg**

**ret**

**rbx t1= 0x14 rax t2 = 0**

**rdx t3 = 0x01**

**while(t3>0){ t2 = t2+t1; t1 = t1-t3;**

**}**

**return 0;**